Introduction to RStudio

STAT 28

January 26, 2017

Introduction to RStudio

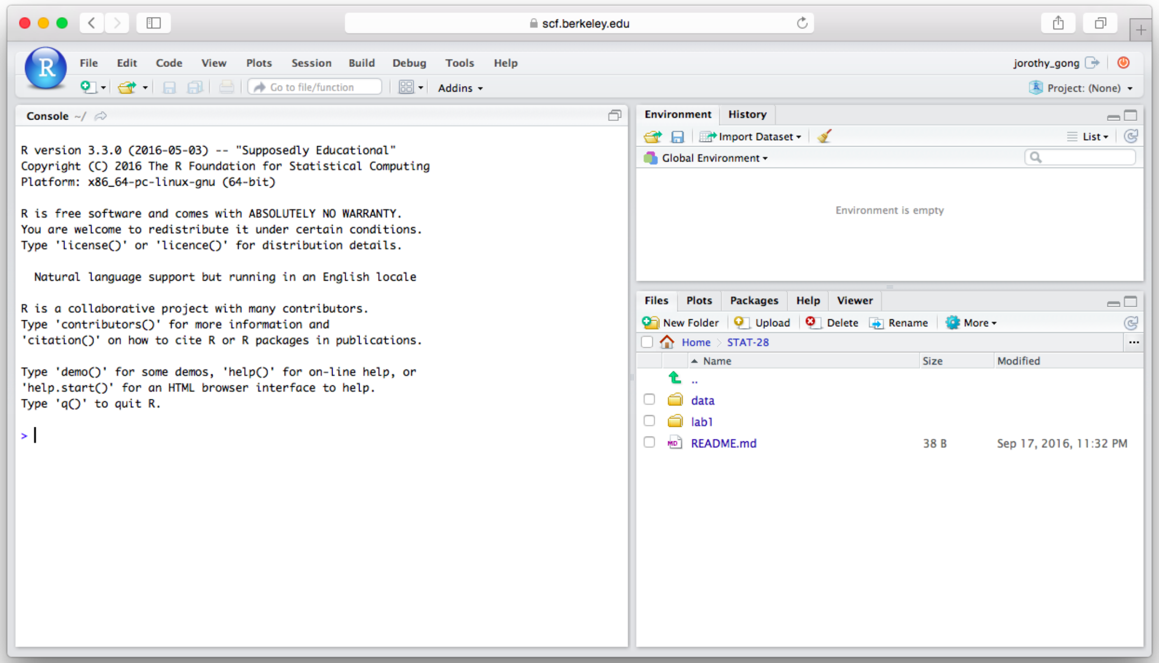
RStudio is an open-source Integrated Development Environment (IDE) which provides many user-friendly features. We will use RStudio throughout the rest of the course. RStudio can be started from the desktop or through the web server. The user interface is generally the same for the two. We will focus on the web server version and provide a brief section on how to install RStudio locally on your own computer.

Web Version

RStudio server can be accessed via a web broswer. All the data storage and computation are done remotely on the server (similar to the web version of ipython notebook that you have experienced in DS 8 and lab 1).

Log In

To login in to the server, open the url https://stat28.berkeley.edu. Once successfully authenticate, you will see the following RStudio interface.



```
R version 3.3.0 (2016-05-03) -- "Supposedly Educational"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> 
```

Name	Size	Modified
..		
data		
lab1		
README.md	38 B	Sep 17, 2016, 11:32 PM

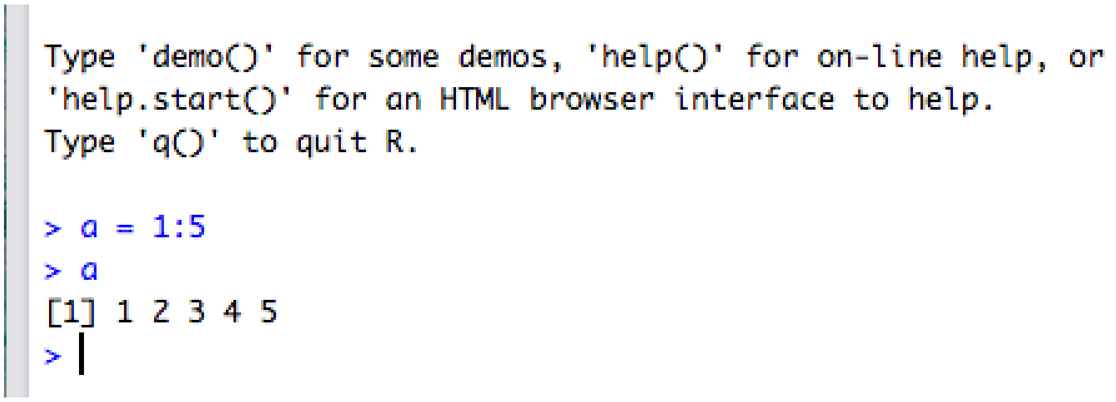
Figure 1: *A screen shot of Rstudio on initial startup*

Your first command in the console

The main window is the "Console" on left panel. The symbol > is a command prompt. And this is where R evaluate your commands. For example, type the following commands in the console:

```
a = 1:5
a
```

```
## [1] 1 2 3 4 5
```



```
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> a = 1:5
> a
[1] 1 2 3 4 5
> 
```

Figure 2: *Your first command in the R console*

R print the outputs (except figures) in the console, right below your commands. You might notice that after you excute the command, vector `a` appears in the upper right `environment` window. In fact, you can always check all the value of objects you created in the `environment` window. Now you may try to explore how to execute commands in the console yourself.

Create an Rscript to store your commands.

R allows you to go back to previous commands using up and down arrow in your keyboard. However, often times you would like to store R commands in a file. To do so, you can create an R Script by clicking `File` -> `New File` -> `R Script` with the menu bar in the top. R will create an R script with the file extention `.R` and the **editor** window will be invoked. R scripts is a file type that stores only R commands and comments.

To run one line or several line of code in an R script, put your cursor in that line or select mulitple lines, and then click the `Run` botton.

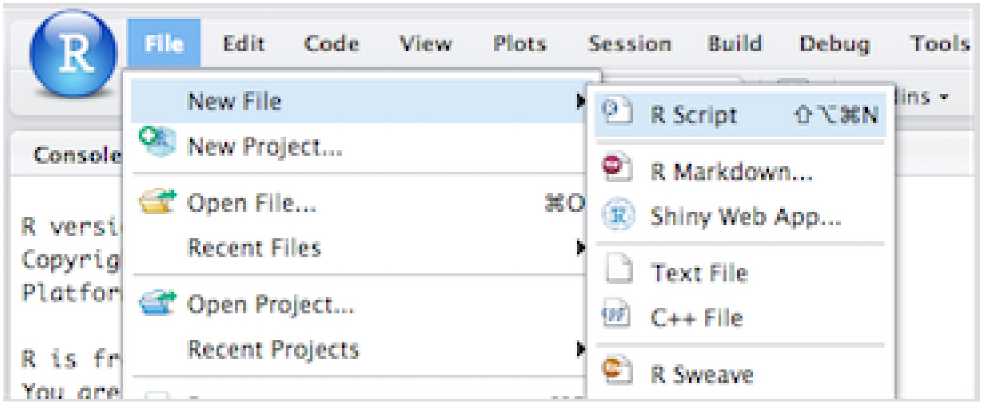
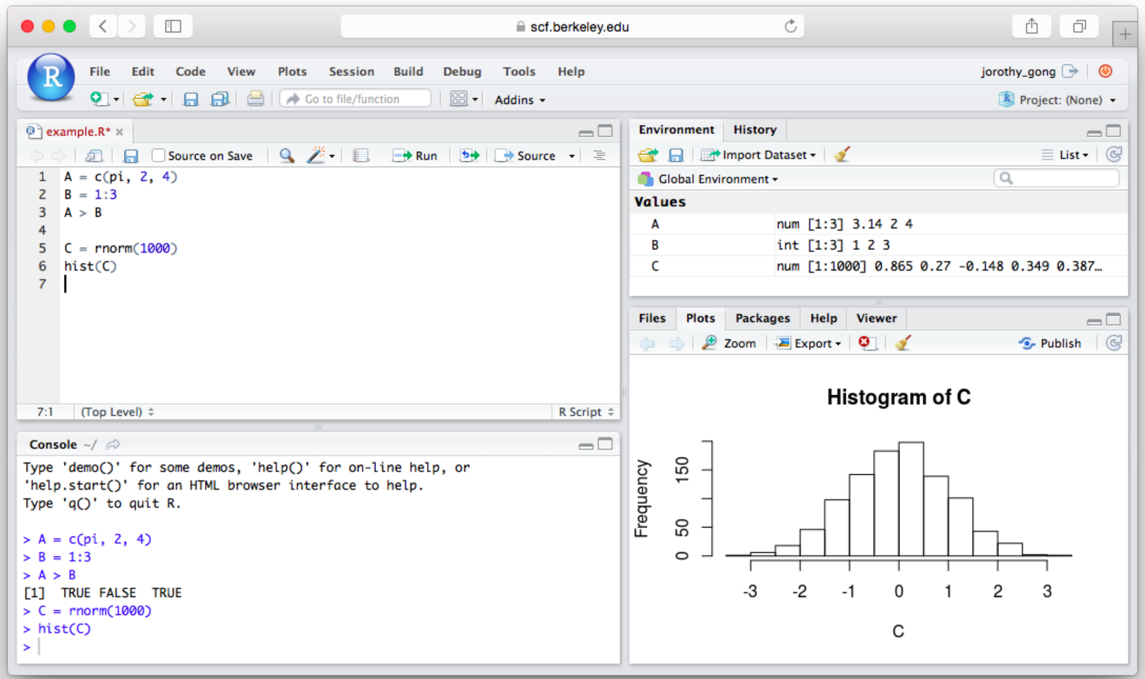


Figure 3: *Create an R script*

RStudio Layout



```
A = c(pi, 2, 4)
B = 1:3
A > B

C = rnorm(1000)
hist(C)
```

```
Values
  A        num [1:3] 3.14 2 4
  B        int [1:3] 1 2 3
  C        num [1:1000] 0.865 0.27 -0.148 0.349 0.387…
```

```
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> A = c(pi, 2, 4)
> B = 1:3
> A > B
[1]  TRUE FALSE  TRUE
> C = rnorm(1000)
> hist(C)
>
```

Figure 4: *A typical Rstudio screen*

The RStudio interface contains several component:

- Bottom left: **console** window. This is where you can type simple command after the the prompt ">". R will excute your current command and return the output in this window.
- Top left: **editor** window. A collection of commands can be edited and saved. This window is hidden on initial startup. You can open it with the menu bar in the top `File -> New File -> R Script`. The editor supports multiple file type, including R script, R markdown and Sweave files. Similarly, existing files can be opened through `File -> Open file ...`. For example, you can open the R script for Lecture 1 `01Probability.R` in lab 2 folder and try to run the code. The red font with * at the end of the file name shows that the current changes are not saved. While you are working in a file, you should save it frequently to avoid losing data unexpectedly.

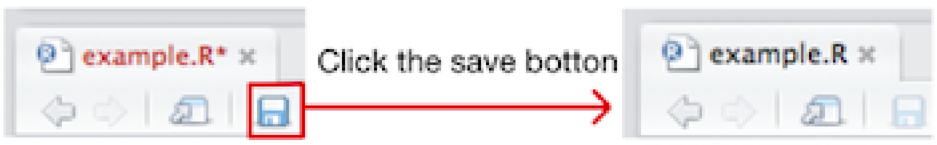
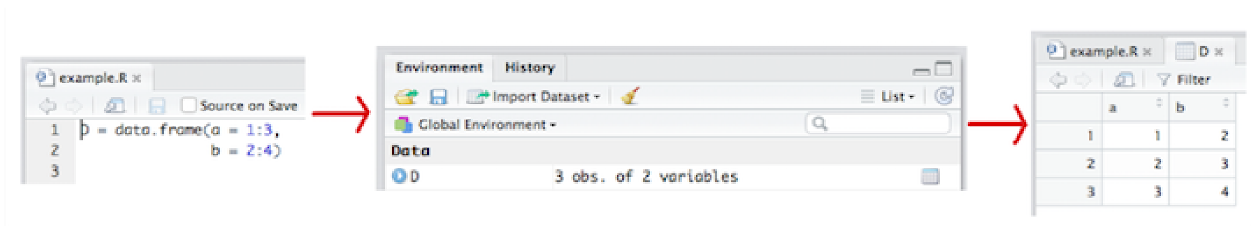


Figure 5:

- Top right: **Environment** / **history** window. In the environment window you can see the data, values and functions you created or loaded. You can view the data by clicking on them. In the example below, if you click the data frame `D` you created, you can see the data on the editor window. The history window shows the commands which has been excuted.


```
D = data.frame(a = 1:3,
               b = 2:4)
```
D   3 obs. of 2 variables

	a	b
1	1	2
2	2	3
3	3	4

Figure 6:

- Bottom right: **files** / **plots** / **packages** / **help** window.
  - **files**: By double clicking the file name, you can open files, then view and edit them in the editor. You can upload files from your own device to the server by clicking the `Upload` botton. To download files from server, check the box before that file and go to `More -> Export`.
  - **plot**: The plots tab will display the graphs. It allows you to go back to pervious plots (by clicking on the left-arrow icon) and export plot as image or pdf.
  - **packages**: install and load packages.
  - **help**: search and view the help document.

Working Directory

Your working directory is the folder on your computer in which you are currently working. When you ask R to open a certain file, it will look in the working directory for this file, and when you tell R to save a data file or figure, it will save it in the working directory.

Before you start working, please set your working directory to where all your data and script files are or should be stored. You can browse and choose working directory by going to `Session -> Set working directory -> Choose directory ...` as follows.

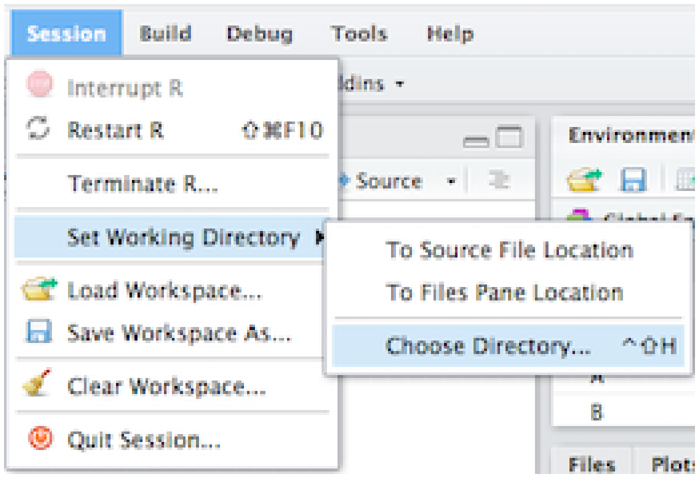


Figure 7:

Packages

R can do many statistical and data analyses. They are organized in packages. To get a list of all installed packages, go to the packages window or type `library()` in the console window. If the box in front of the package name is ticked, the package is loaded (activated) and can be used.

- To install a package. For example, if we want to install a package named `ggplot2`, click install packages in the packages window and type ggplot to search and install. Or you can type `install.packages("ggplot2")` in the console.
- To load a package. To use a installed package, you have to load it to you current working environment. You can check box in front of geometry or type library("geometry") in the command window.

Reading and Saving Files

Similar to Python, R can read/write data files from/to txt or csv.

```
# Read csv files
twitter_follows <- read.csv("twitter_follows.csv",
                            stringsAsFactors = FALSE)
# Read txt files, and treat the first row of the txt file as the column names (header)
twitter_follows <- read.table("twitter_follows.txt",
                              header = TRUE,
                              stringsAsFactors = FALSE)
```

1. What will happen if you set `header to FALSE` or set string?
2. In most cases, we would like to set `stringsAsFactors = FALSE`. `stringsAsFactors = FALSE` tells R to keep character variables as charactor rather than convert to factors.

```
# Write a data frame to csv
write.csv(twitter_follows,
          file = "twitter_follows.csv",
          row.names = FALSE)
# Write a data frame to txt
write.table(twitter_follows,
            file = "twitter_follows.txt",
            row.names = FALSE,
            quote = FALSE)
```

Questions:

1. What will happen if you set `row.names = TRUE` or `quote = TRUE`?

R provide another method for saving and loading R data. You can save(load) several R objects to(from) `.RData` file at the same time. And load them when you start a new R session in the future.

```
x <- 1:3
y <- list(a = 1, b = TRUE, c = "oops")
# save x, y
save(x, y, file = "xy.RData")
# load x, y
load("xy.RData")
```

R provides methods to save the environment (all data, values, functions, etc.) in RData for future use. You could excute the code as follows, or click the save botton in the environment window to save and click the `RData` files in the file window to load them.

```
# save the environment
save.image("~/foo.RData")
# load the environment
load("~/foo.RData")
```

Desktop Version

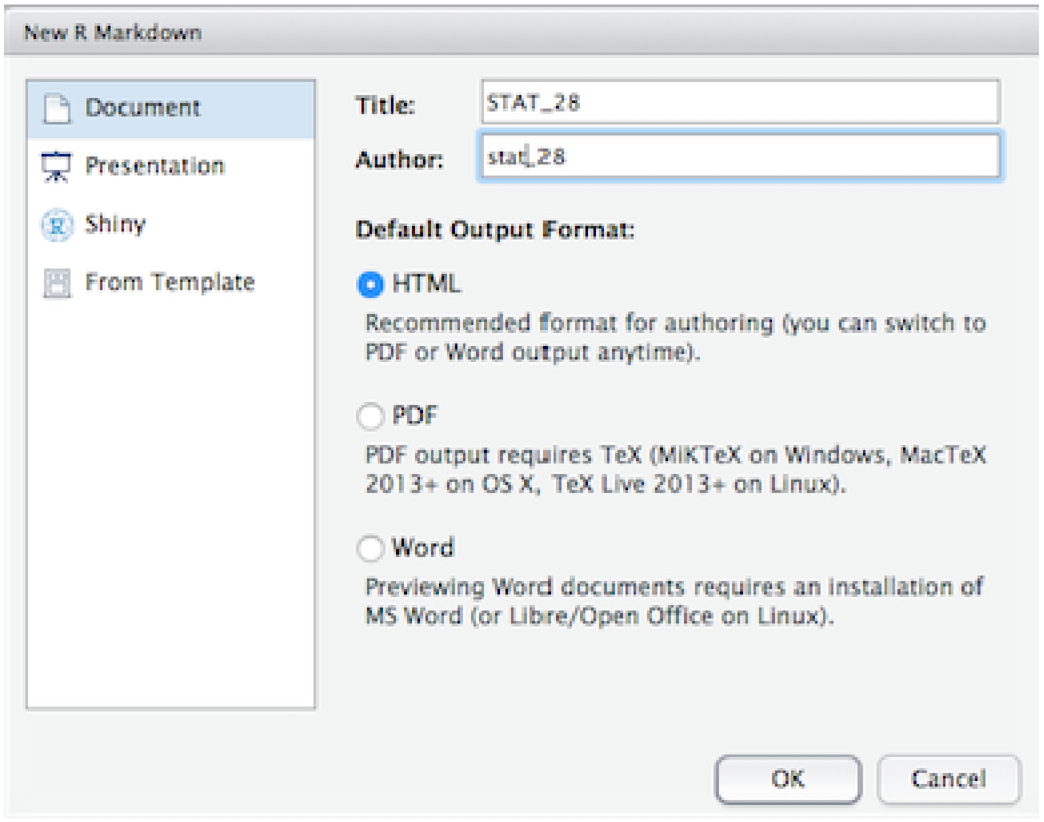
The layout looks similar to that of the web version. The installation of R and RStudio is straightforward, just as installing other applications in your computer.

- To install R on your computer (legally for free!), go to the home website of R https://cran.r-project.org/ and choose the platform (Mac, Windows or Linux) corresponds to your own divice.
- After successfully installed R, you can https://www.rstudio.com/products/rstudio/download3/ according to your platform.

Introduciton to R Markdown

The relationship between R Scipts and R Markdown is similar to the relationship between Python scripts and IPython Notebooks. While R Scipts contains only R code, R Markdown enable us to create dynamic documents with embedded chunks of R code. The document is then converted to other formats such as HTML and PDF which is easy to share. Unlike IPython notebook in which chunks are run and results are printed seperately, R Markdown automatically run all the code by order when generating HTML or PDF files. This feature makes it fully reproducible for others.

To create an R Markdown file, click on `File` -> `New File` -> `R Markdown`. You should now see a dialog as shown below. Select 'Document' in the left panel and fill in title and author field and hit 'OK'.


New R Markdown

Title: STAT_28

Author: stat_28

Default Output Format:

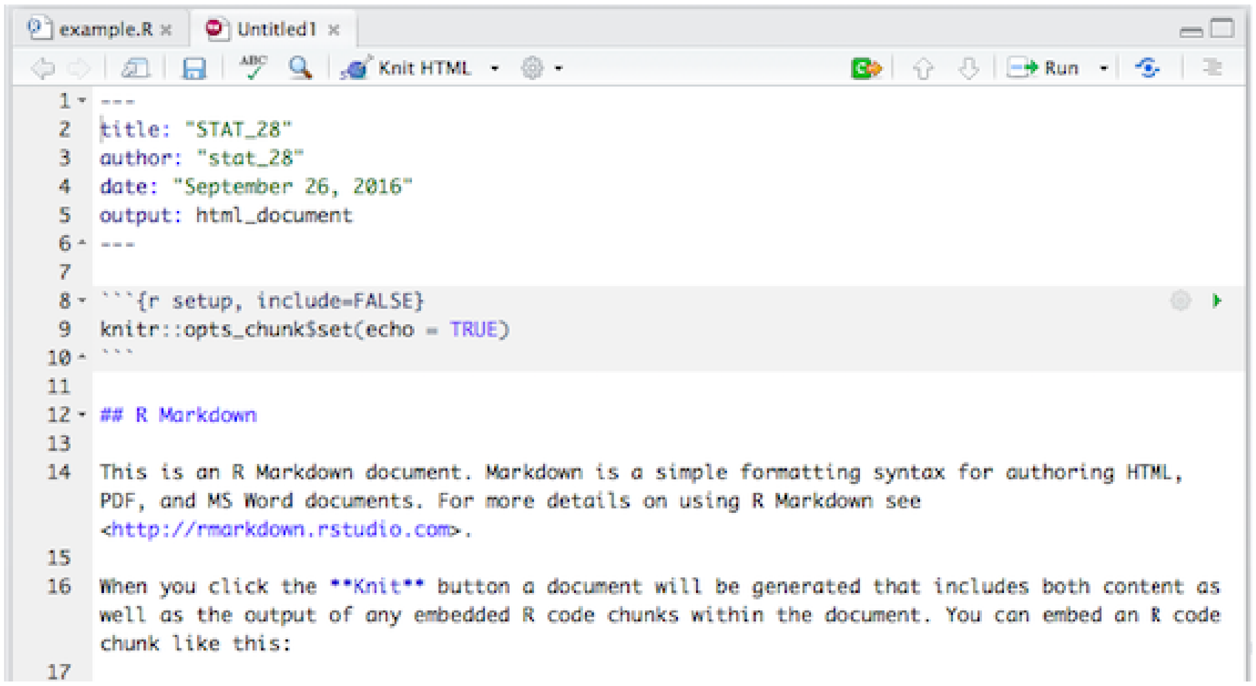
[✓ HTML]
Recommended format for authoring (you can switch to PDF or Word output anytime).

PDF
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

Word
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

Figure 8:

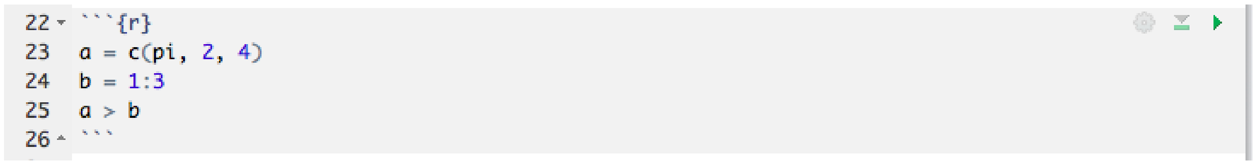
You should now have a document which looks like this:



~~~
---
title: "STAT_28"
author: "stat_28"
date: "September 26, 2016"
output: html_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:
~~~

Figure 9:

To embed R code into the document, click on the add chunk icon . It should then add a code chunk in the document, you may write code in the chunk as you do in IPython notebooks.

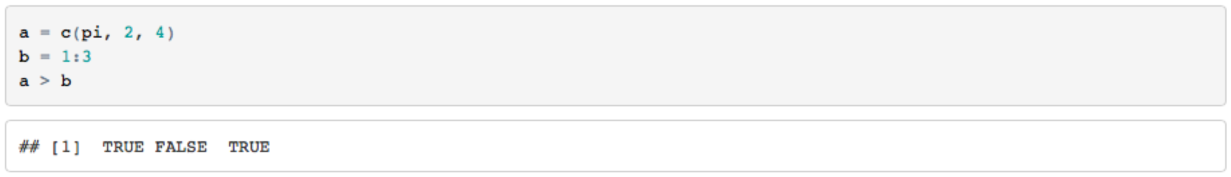


~~~
```{r}
a = c(pi, 2, 4)
b = 1:3
a > b
```
~~~

Figure 10:

To generate an HTML file, click on the icon Knit HTML. It should generate the HTML in the same folder as the R Markdown file.

The above code chunk will look as follows in the HTML file:

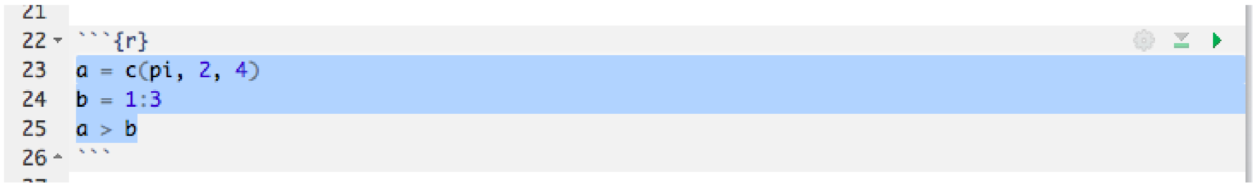


```
a = c(pi, 2, 4)
b = 1:3
a > b
```

```
## [1]  TRUE FALSE  TRUE
```

Figure 11:

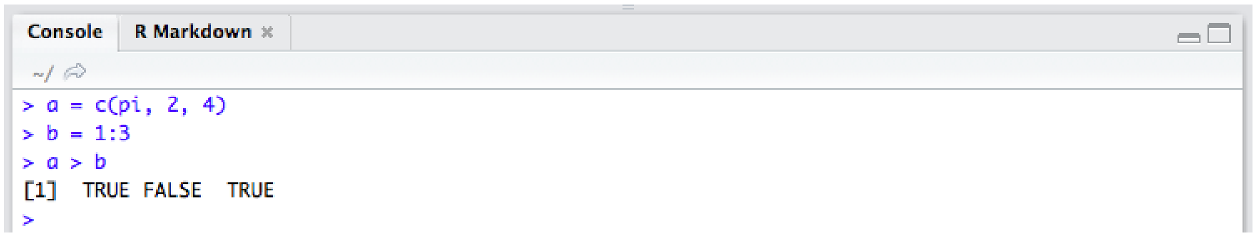
Similar as in R script file, you can run one or several lines of code from the R Markdown. To run several lines of code, You may select a piece of code as follows:



~~~
```{r}
a = c(pi, 2, 4)
b = 1:3
a > b
```
~~~

Figure 12:

Then if you click on the **run** icon in the editor window, RStudio will run the code and give the output in the console which look like this:



```
> a = c(pi, 2, 4)
> b = 1:3
> a > b
[1]  TRUE FALSE  TRUE
>
```

Figure 13:

Similarly, if you would like to run a single line of code. Put the cursor in that line and click **run**.

Please refer to an introduction of Markdown Basics on how to add headers, lists and images on the markdown file.

Reference

- Getting Started with RStudio, CHAPTER 1, Overview, Installation
- Torfs, P and Brauer, C. A (very) short introduction to R
- Oscar Torres-Reyna. Introduction to RStudio
- R Mardown tutorial from RStudio
- R-Markdown and Knitr Tutorial